US009251221B1

(12) **United States Patent**
Murphey et al.

(10) **Patent No.:** **US 9,251,221 B1**
(45) **Date of Patent:** **Feb. 2, 2016**

(54) **ASSIGNING SCORES TO OBJECTS BASED ON SEARCH QUERY RESULTS**

(71) Applicant: **Splunk Inc.**, San Francisco, CA (US)

(72) Inventors: **Lucas Murphey**, Wadsworth, IL (US); **David Hazekamp**, Tinley Park, IL (US)

(73) Assignee: **Splunk Inc.**, San Francisco, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 18 days.

(21) Appl. No.: **14/447,995**

(22) Filed: **Jul. 31, 2014**

Related U.S. Application Data

(60) Provisional application No. 62/027,239, filed on Jul. 21, 2014.

(51) **Int. Cl.**
    G06F 15/173 (2006.01)
    G06F 17/30 (2006.01)
(52) **U.S. Cl.**
    CPC *G06F 17/3053* (2013.01)
(58) **Field of Classification Search**
    CPC H04L 43/05; G06F 17/3053
    See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

8,402,012 B1 *	3/2013	Herz et al.	....	707/708
8,412,696 B2	4/2013	Zhang et al.		
8,589,403 B2	11/2013	Marquardt et al.		
8,682,925 B1	3/2014	Marquardt et al.		
8,744,894 B2 *	6/2014	Christiansen et al.	....	705/7.28

2009/0328222 A1 *	12/2009	Helman et al.	....	726/25
2011/0265162 A1 *	10/2011	Alavandar et al.	....	726/7
2013/0318236 A1 *	11/2013	Coates et al.	....	709/224
2013/0318603 A1	11/2013	Merza		
2013/0326620 A1	12/2013	Merza et al.		
2014/0160238 A1	6/2014	Yim et al.		

OTHER PUBLICATIONS

Bitincka, Ledion, et al., "Optimizing Data Analysis with a Semi-Structured Time Series Database", Splunk Inc., 2010 pp. 1-9.
Carasso, David, "Exploring Splunk Search Processing Language (SPL) Primer and Cookbook", Splunk Inc., 2012 CITO Research, New York, 154 Pages.
http://docs.splunk.com/Documentation/PCI/2.1.1/ [000119] User/IncidentReviewdashboard, 2 Pages (Last accessed Aug. 5, 2014).
"vSphere Monitoring and Performance", VMware, Inc., Update 1, vSphere 5.5, EN-001357-02, 2010-2014, pp. 1-174 http://pubs.vmware.com/ vsphere-55/topic/com.vmware.ICbase/PDF/vsphere-esxi-vcenter-server-551-monitoring-performance-guide.pdf.
U.S. Appl. No. 14/167,316, filed Jan. 29, 2014.
U.S. Appl. No. 14/266,812, filed Apr. 30, 2014.
U.S. Appl. No. 14/266,817, filed Apr. 30, 2014.

* cited by examiner

Primary Examiner — Tarek Chbouki
(74) *Attorney, Agent, or Firm* — Perkins Coie LLP

(57) **ABSTRACT**

Systems and methods for assigning scores to objects based on evaluating triggering conditions applied to datasets produced by search queries in data aggregation and analysis systems. An example method may comprise: executing, by one or more processing devices, a search query to produce a dataset comprising one or more data items derived from source data; and responsive to determining that at least a portion of the dataset satisfies a triggering condition, modifying a score assigned to an object to which the portion of the dataset pertains.

30 Claims, 17 Drawing Sheets

100

SEARCH NAME [ACTIVITY FROM EXPIRED USER IDENTITY] ⟍ 107

APPLICATION CONTEXT [DA-ESS-IDENTITYMANAGEMENT ⇕] ⟍ 115

DESCRIPTION [ALERTS WHEN AN EVENT IS DISCOVERED FROM A USER A] ⟍ 110

SEARCH
```
| DATAMODEL "IDENTITY_MANAGEMENT"
"EXPIRED_IDENTITY_ACTIVITY" SEARCH | STATS
MAX(_TIME) AS "LASTTIME",LATEST(_RAW) AS
"ORIG_RAW", COUNT BY
"EXPIRED_IDENTITY_ACTIVITY.EXPIRED_USER"
| RENAME "EXPIRED_IDENTITY_ACTIVITY.EXPIRED_USER"
AS "USER" | EVAL RISK_SCORE=COUNT*10|
```
⟍ 120

EDIT SEARCH IN GUIDED MODE

TIME RANGE

START TIME [RT-5M@M] ⟋ 125A

END TIME [RT+5M@M] ⟋ 125B

CRON SCHEDULE* [*/5****] ⟋ 130

THROTTLING

WINDOW DURATION [86300] ⟋ 135

FIELDS TO GROUP BY [USER ✕] ⟍ 140

RISK SCORING

CREATE RISK MODIFIER ☑ ⟋ 145

SCORE* [80] ⟋ 150

RISK OBJECT FIELD* [USER] ⟋ 155

RISK OBJECT TYPE* [USER ▼] ⟋ 160

ACTIONS

INCLUDE IN RSS FEED ☐ ⟋ 165A

SEND EMAIL ☐ ⟋ 165B

RUN A SCRIPT ☐ ⟋ 165C

FIG. 1

FIG. 2

RECENT RISK MODIFIERS

TIME	RISK_OBJECT	RISK_OBJECT_TYPE	SOURCE	DESCRIPTION	RISK_SCORE <1m AGO
2014-07-02 10:57:39	PROD-POS-003	SYSTEM	ACCESS - EXCESSIVE FAILED LOGINS - RULE	DETECTS EXCESSIVE NUMBER OF FAILED LOGIN ATTEMPT (THIS IS LIKELY A BRUTE FORCE ATTACK)	60
2014-07-02 10:57:38	SE-003	SYSTEM	ACCESS - INSECURE OR CLEARTEXT AUTHENTICATION - RULE	DETECTS AUTHENTICATION REQUESTS THAT TRANSMIT THE PASSWORD OVER THE NETWORK AS CLEARTEXT (UNENCRYPTED)	80
2014-07-02 10:57:38	PROD-MFS-006	SYSTEM	ACCESS - INSECURE OR CLEARTEXT AUTHENTICATION - RULE	DETECTS AUTHENTICATION REQUESTS THAT TRANSMIT THE PASSWORD OVER THE NETWORK AS CLEARTEXT (UNENCRYPTED)	80
2014-07-02 10:57:38	COREDEV-005	SYSTEM	ACCESS - INSECURE OR CLEARTEXT AUTHENTICATION - RULE	DETECTS AUTHENTICATION REQUESTS THAT TRANSMIT THE PASSWORD OVER THE NETWORK AS CLEARTEXT (UNENCRYPTED)	80
2014-07-02 10:57:36	PROD-MFS-006	SYSTEM	ACCESS - BRUTE FORCE ACCESS BEHAVIOR DETECTED - RULE	DETECTS EXCESSIVE NUMBER OF FAILED LOGIN ATTEMPTS ALONG WITH A SUCCESSFUL ATTEMPT (THIS COULD INDICATE A SUCCESSFUL BRUTE FORCE ATTACK)	80
2014-07-02 10:57:35	ACME-004	SYSTEM	ENDPOINT - HIGH OR CRITICAL PRIORITY HOST WITH MALWARE - RULE	ALERTS WHEN AN INFECTION IS NOTED ON A HOST WITH HIGH OR CRITICAL PRIORITY.	80
2014-07-02 10:57:35	ACME-001	SYSTEM	ENDPOINT - HIGH OR CRITICAL PRIORITY HOST WITH MALWARE - RULE	ALERTS WHEN AN INFECTION IS NOTED ON A HOST WITH HIGH OR CRITICAL PRIORITY.	80
2014-07-02 10:57:34	HOST-002	SYSTEM	ENDPOINT - HIGH OR CRITICAL PRIORITY HOST WITH MALWARE - RULE	ALERTS WHEN AN INFECTION IS NOTED ON A HOST WITH HIGH OR CRITICAL PRIORITY.	80
2014-07-02 10:57:34	PROD-POS-005	SYSTEM	AUDIT - ANOMALOUS AUDIT TRAIL ACTIVITY DETECTED - RULE	DISCOVERS ANOMALOUS ACTIVITY SUCH AS THE DELETION OF OR CLEARING OF LOG FILES. ATTACKERS OF OFTENTIMES CLEAR THE LOG FILES IN ORDER TO HIDE THEIR ACTIONS; THEREFORE, THIS MAY INDICATE THAT THE SYSTEM HAS BEEN COMPROMISED.	40
2014-07-02 10:57:12	ACME-001	SYSTEM	ACCESS - EXCESSIVE FAILED LOGINS - RULE	DETECTS EXCESSIVE NUMBER OF FAILED LOGIN ATTEMPTS (THIS IS LIKELY A BRUTE FORCE ATTACK)	60

FIG. 2 (Cont.)

300A

START

310 Execute search query
associated with alert

315
Triggering condition
satisfied?

no

yes

320 Modify score value of primary
object

330 Modify score values of objects
associated with primary object

END

FIG. 3A

300B

START

350 Sort score values associated with user selected objects

355 Cause score values to be displayed

360 User selection of score modifier received?

no

yes

365 Cause information pertaining to score modifier to be displayed

END

FIG. 3B

FIG. 4

```
                    ┌─────────┐
                    │  START  │
                    └────┬────┘
                         │
                         ▼
          ┌────────────────────────────┐
          │       RECEIVE DATA          │
          │           201               │
          └──────────────┬─────────────┘
                         │
                         ▼
          ┌────────────────────────────┐
          │  APPORTION DATA INTO EVENTS │
          │           202               │
          └──────────────┬─────────────┘
                         │
                         ▼
          ┌────────────────────────────┐
          │  DETERMINE TIMESTAMPS FROM  │
          │           EVENTS            │
          │           203               │
          └──────────────┬─────────────┘
                         │
                         ▼
          ┌────────────────────────────┐
          │  ASSOCIATE TIMESTAMPS WITH  │
          │           EVENTS            │
          │           204               │
          └──────────────┬─────────────┘
                         │
                         ▼
          ┌────────────────────────────┐
          │      TRANSFORM EVENTS       │
          │           205               │
          └──────────────┬─────────────┘
                         │
                         ▼
          ┌ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ┐
          │  IDENTIFY KEYWORDS IN EVENTS │
          │           206               │
          └ ─ ─ ─ ─ ─ ─ ─┬─ ─ ─ ─ ─ ─ ─┘
                         │
                         ▼
          ┌ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ┐
          │     UPDATE KEYWORD INDEX     │
          │           207               │
          └ ─ ─ ─ ─ ─ ─ ─┬─ ─ ─ ─ ─ ─ ─┘
                         │
                         ▼
          ┌────────────────────────────┐
          │  STORE EVENTS IN DATA STORE │
          │           208               │
          └──────────────┬─────────────┘
                         │
                         ▼
                    ┌─────────┐
                    │   END   │
                    └─────────┘
```

FIG. 5

START

SEARCH HEAD RECEIVES QUERY
FROM CLIENT
301

SEARCH HEAD DETERMINES WHAT
PORTIONS OF THE QUERY CAN BE
DISTRIBUTED TO INDEXERS
302

SEARCH HEAD DISTRIBUTES
PORTIONS OF QUERY TO
INDEXERS
303

INDEXERS SEARCH DATA STORE
FOR QUERY-RESPONSIVE EVENTS
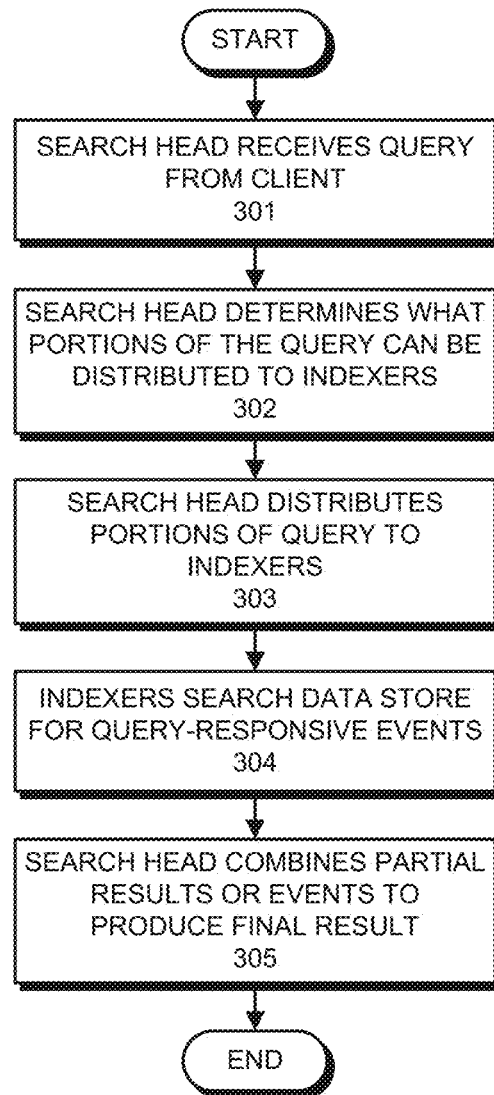304

SEARCH HEAD COMBINES PARTIAL
RESULTS OR EVENTS TO
PRODUCE FINAL RESULT
305

END

FIG. 6

FIG. 7

```
Original Search: 501
search "error | stats count BY host


Sent to peers: 502
search "error | prestats count BY host (map)


Executed by search head: 503
Merge prestats results received from peers (reduce)
```

FIG. 8

SEARCH SCREEN 600

SEARCH PIVOT REPORTS ALERTS DASHBOARDS SEARCH & REPORTING

Q NEW SEARCH SAVE AS MENU → SAVE AS ˅ CLOSE

BUTTER CUP GAMES SEARCH BAR 602 TIME RANGE PICKER 612 → ALL TIME ˅ Q

✓ 36,819 EVENTS (BEFORE 4/30/14 2:19:02.000 PM) SEARCH RESULTS TABS 604 JOB ˅ II ▢ ↱ ⊕ ↓ ⌂ ᴾ SMART MODE ˅

EVENTS (36,819) STATISTICS VISUALIZATION SEARCH ACTION BUTTONS SEARCH MODE SELECTOR

FORMAT TIMELINE ˅ -ZOOM OUT +ZOOM TO SELECTION ×DESELECT TIMELINE 605 1 HOUR PER COLUMN

LIST ˅ FORMAT ˅ 20 PER PAGE ˅ <PREV 1 2 3 4 5 6 7 8 9 … NEXT>

< HIDE FIELDS ≡ ALL FIELDS

	TIME	EVENT
	4/28/14 6:22:16.000 PM	91.205.189.15 - - [28/APR/2014:18:22:16] "GET /OLDLINK?ITEMID=EST-14&JSESSIONID=SD6SL7FF 7ADFF53113 HTTP 1.1" 200 1665 "HTTP://WWW.BUTTERCUPGAMES.COM/OLDLINK?ITEMID=EST-14" "MOZ ILLA/5.0 (WINDOWS NT 6.1; WOW64) APPLEWEBKIT/536.5 (KHTML, LIKE GECKO) CHROME/19.0.1084. 46 SAFARI/536.5" 159
		HOST = WWW2 ⁝ SOURCE = TUTORIALDATA.ZIP:.//WWW2/ACCESS.LOG ⁝ SOURCETYPE = ACCESS_COMBINED_WCOOKIE
	4/28/14 6:20:56.000 PM	182.236.164.11 - - [28/APR/2014:18:20:56] "GET /CART.DO?ACTION=ADDTOCART&ITEMID=EST-15&P RODUCTID=BS-AG-G09&JSESSIONID=SD6SL8FF10ADFF53101 HTTP 1.1" 200 2252 "HTTP://WWW.BUTTERC UPGAMES.COM/OLDLINK?ITEMID=EST-15" "MOZILLA/5.0 (MACINTOSH; INTEL MAC OS X 10_7_4) APPLE WEBKIT/536.5 (KHTML, LIKE GECKO) CHROME/19.0.1084.46 SAFARI/536.5" 506
		HOST = WWW1 ⁝ SOURCE = TUTORIALDATA.ZIP:.//WWW1/ACCESS.LOG ⁝ SOURCETYPE = ACCESS_COMBINED_WCOOKIE
	4/28/14 6:20:55.000 PM	182.236.164.11 - - [28/APR/2014:18:20:55] "POST /OLDLINK?ITEMID=EST-18&JSESSIONID=SD6SL8 FF10ADFF53101 HTTP 1.1" 408 893 "HTTP://WWW.BUTTERCUPGAMES.COM/PRODUCT.SCREEN?PRODUCTID= SF-BVS-G01" "MOZILLA/5.0 (MACINTOSH; INTEL MAC OS X 10_7_4) APPLEWEBKIT/536.5 (KHTML, LI KE GECKO) CHROME/19.0.1084.46 SAFARI/536.5" 134
		HOST = WWW1 ⁝ SOURCE = TUTORIALDATA.ZIP:.//WWW1/ACCESS.LOG ⁝ SOURCETYPE = ACCESS_COMBINED_WCOOKIE

EVENTS LIST 608

FIELDS SIDEBAR 606

SELECTED FIELDS

a HOST 3
a SOURCE 3
a SOURCETYPE 1

INTERESTING FIELDS

a ACTION 5
BYTES 100+
a CATEGORYID 8
a CLIENTIP 100+
DATE_HOUR 24
DATE_MDAY 8
DATE_MINUTE 60

FIG. 9A

DATA SUMMARY   ✗

HOSTS (5) \ SOURCES (8) \ SOURCETYPES (3)

(FILTER)

HOST ◊		COUNT ◊	LAST UPDATE ◊
MAILSV	✎∨	9,829	4/29/14 1:32:47.000 PM
VENDOR_SALES	✎∨	30,244	4/29/14 1:32:46.000 PM
WWW1	✎∨	24,221	4/29/14 1:32:44.000 PM
WWW2	✎∨	22,595	4/29/14 1:32:47.000 PM
WWW3	✎∨	22,975	4/29/14 1:32:45.000 PM

FIG. 9B

KEY INDICATOR VIEW 700

ACCESS NOTABLES	ENDPOINT NOTABLES	NETWORK NOTABLES	IDENTITY NOTABLES	AUDIT NOTABLES
TOTAL COUNT	TOTAL COUNT	TOTAL COUNT	TOTAL COUNT	TOTAL COUNT
45 ⬈ +45	**61** ⬈ +61	**15** ⬈ +15	**2** ⬈ +2	**32** ⬈ +32

MALWARE INFECTIONS	VULNERABLE HOSTS	VULNERABILITIES / HOST AVG	HOSTS FULLY PATCHED
TOTAL COUNT	TOTAL COUNT	MEDIUM SEVERITY OR HIGHER	PROCESS OF TOTAL HOSTS
632 ⬈ +63	**1452** ⬈ -74	**1.6** ⬊ -0.2	**78.3%** ⬈ +0.2

701
702
702
703

NOTABLE EVENTS BY URGENCY

14M AGO

URGENCY

UNKNOWN
INF...NAL
LOW
MEDIUM
HIGH

NOTABLE EVENTS BY TIME

EVENTS

20

10

12:00 PM 12:00 AM 12:00
TUE   WED   PM
JUL 16   JUL 17

— ACCESS
— AUDIT
— — ENDPOINT
· · · · IDENTITY
— — NETWORK
– · – THREAT

FIG. 10A

INCIDENT REVIEW DASHBOARD 710

INCIDENT REVIEW | ACTIONS˅

STATUS: URGENCY: OWNER: TITLE: INCIDENT
                           HIGH   ATTRIBUTES
SECURITY DOMAIN: GOVERNANCE: SEARCH: FIELDS
                           PCI   711
🔍

TIME RANGE FIELD 712

☑ 225 MATCHING EVENTS   🖫 SAVE ▾ 🔍 CREATE ▾

☑ HIDE 🔍 ZOOM OUT 🔍 ZOOM TO SELECTION [i] DESELECT   LINEAR SCALE ▾ 1 BAR = 1 HOUR

120   TIMELINE 713   120
60   60

4:00 AM   6:00 AM   8:00 AM
SUN AUG 26 2012

225 EVENTS IN A 24 HOUR WINDOW (REAL-TIME) (FROM 11:29:20 AM AUGUST 25 TO 11:29:20 AM AUGUST 26,2012   EVENTS LIST 714
        << PREV 1 2 3 4 5 6 7 8 9 10 NEXT >> | EDIT SELECTED EVENTS | EDIT ALL

SELECT ALL | UNSELECT ALL

24 HOUR WINDOW ▾
LAST 15 MINUTES
LAST 60 MINUTES
LAST 4 HOURS
LAST 24 HOURS
LAST 7 DAYS
LAST 30 DAYS
LAST YEAR
REAL-TIME
OTHER
ALL TIME
CUSTOM TIME.

SELECT	OPTIONS	TIME	SECURITY DOMAIN	TITLE	URGENCY	STATUS	OWNER
☐	▶	8/26/12 11:11:03.000 AM	ACCESS ▾	INSECURE OR CLEARTEXT AUTHENTICATION DETECTED ▾	① HIGH ▾	NEW ▾	UNASSIGNED ▾ VIEW DETAILS
☐	▶	8/26/12 11:10:07.000 AM	ACCESS ▾	INSECURE OR CLEARTEXT AUTHENTICATION DETECTED ▾	① HIGH ▾	NEW ▾	UNASSIGNED ▾ VIEW DETAILS
☐	▶	8/26/12 11:00:39.000 AM	ACCESS ▾	ACCOUNT (BLINEBRY) DELETED ON (PROD-POS-001) ▾	① HIGH ▾	NEW ▾	UNASSIGNED ▾ VIEW DETAILS
☐	▶	8/28/12 11:00:39.000 AM	ACCESS ▾	ACCOUNT (BEU) DELETED ON (COREDEV-006) ▾	① HIGH ▾	NEW ▾	UNASSIGNED ▾ VIEW DETAILS
☐	▶	8/28/12 11:00:39.000 AM	ACCESS ▾	ACCOUNT (COMBS) DELETED ON (HOST-005) ▾	① HIGH ▾	NEW ▾	UNASSIGNED ▾ VIEW DETAILS
☐	▶	8/28/12	ACCESS ▾	ACCOUNT (WISNER) DELETED ON (BUSDEV-005) ▾	① HIGH ▾	NEW ▾	UNASSIGNED ▾ VIEW DETAILS

FIG. 10B

FIG. 10C

RECENT TASKS AND EVENTS
« PREV ☐ 2 3 4 5 NEXT »

	TIME ◆	MESSAGE ◆
1	11/20/13 1:45:21.200 PM	ALARM 'HOST CPU USAGE' ON APPS-ESX501.SV.SPLUNK.COM CHA...
2	11/20/13 1:44:37.350 PM	CHANGED RESOURCE ALLOCATION FOR LOAD-VM-18
3	11/20/13 1:44:31.810 PM	MIGRATING LOAD-VM-18 OFF HOST APPS-ESX501.SV.SPLUNK.COM
4	11/20/13 1:44:31.470 PM	MIGRATING LOAD-VM-18 FROM APPS-ESX501.SV.SPLUNK.COM. ISC...
5	11/20/13 1:44:28.100 PM	TASK: MIGRATE VIRTUAL MACHINE

RECENT ESXi LOG ENTRIES
« PREV ☐ 2 3 4 5 6 7 8 9 10 11 12 13 14 15 NEXT »

	TIME ◆	HOST ◆	MESSAGE ◆
1	11/20/2013 13:50:14.381	APPS-ESX501.SV.SPLUNK.COM	IVPX(RCI-FINISH TASK-...
2	11/20/2013 13:50:14.229	APPS-ESX501.SV.SPLUNK.COM	IVPX(RCI-BEGIN TASK-...
3	11/20/2013 13:50:08.325	APPS-ESX501.SV.SPLUNK.COM	IVPX(RCI-FINISH TASK-...
4	11/20/2013 13:50:08.310	APPS-ESX501.SV.SPLUNK.COM	IVPX(RCI-BEGIN TASK-...
5	11/20/2013 13:50:05.310	APPS-ESX501.SV.SPLUNK.COM	IVPX(RCI-FINISH TASK-...

CPU ▼   AGGREGATED ▼   AVERAGE CPU CORE UTILIZATION ▼   AVERAGE ▼

PULL-DOWN MENUS 742

DISPLAY MARKERS NO ◆

P AVERAGE_CPU
CORE UTILIZATION_
PERCENT

— CRITICAL

— WARNING

125

100

75

50

25

10:00 AM   11:00 AM   12:00 PM   1:00 PM
WED NOV 20
2013

TIME

FIG. 10D

COMPUTING DEVICE 1000

PROCESSING DEVICE 1002

ASSIGNING SCORE
VALUES 300A-300B

VIDEO DISPLAY
1010

MAIN MEMORY 1004

ASSIGNING SCORE
VALUES 300A-300B

ALPHA-NUMERIC
INPUT DEVICE
1012

1030

STATIC MEMORY
1006

CURSOR CONTROL
DEVICE
1014

NETWORK INTERFACE
DEVICE
1008

SIGNAL GENERATION
DEVICE
1016

NETWORK
1020

DATA STORAGE DEVICE 1018

COMPUTER-READABLE
STORAGE MEDIUM 1028

ASSIGNING SCORE
VALUES 300A-300B

FIG. 11

ASSIGNING SCORES TO OBJECTS BASED ON SEARCH QUERY RESULTS

RELATED APPLICATIONS

This application claims the priority benefit of U.S. Provisional Patent Application No. 62/027,239 filed on Jul. 21, 2014.

TECHNICAL FIELD

The present disclosure is generally related to data aggregation and analysis systems, and is more specifically related to assigning scores to objects based on evaluating triggering conditions applied to datasets produced by search queries.

BACKGROUND

Modern data centers often comprise thousands of hosts that operate collectively to service requests from even larger numbers of remote clients. During operation, components of these data centers can produce significant volumes of machine-generated data. The unstructured nature of much of this data has made it challenging to perform indexing and searching operations because of the difficulty of applying semantic meaning to unstructured data. As the number of hosts and clients associated with a data center continues to grow, processing large volumes of machine-generated data in an intelligent manner and effectively presenting the results of such processing continues to be a priority.

BRIEF DESCRIPTION OF THE DRAWINGS

The present disclosure is illustrated by way of examples, and not by way of limitation, and may be more fully understood with references to the following detailed description when considered in connection with the figures, in which:

FIG. **1** schematically illustrates an example GUI for specifying security score modification rules, including search queries, triggering conditions, and other information to be utilized by the system for assigning and/or modifying security risk scores associated with various objects, in accordance with one or more aspects of the present disclosure;

FIG. **2** schematically illustrates an example GUI for visually presenting security risk scores assigned to a plurality of objects, in accordance with one or more aspects of the present disclosure;

FIGS. **3A-3B** depict flow diagrams of example methods **300A-300B** for assigning scores to objects based on evaluating triggering conditions applied to datasets produced by search queries, in accordance with one or more aspects of the present disclosure;

FIG. **4** presents a block diagram of an event-processing system in accordance with one or more aspects of the present disclosure;

FIG. **5** presents a flowchart illustrating how indexers process, index, and store data received from forwarders in accordance with one or more aspects of the present disclosure;

FIG. **6** presents a flowchart illustrating how a search head and indexers perform a search query in accordance with one or more aspects of the present disclosure;

FIG. **7** presents a block diagram of a system for processing search requests that uses extraction rules for field values in accordance with one or more aspects of the present disclosure;

FIG. **8** illustrates an exemplary search query received from a client and executed by search peers in accordance with one or more aspects of the present disclosure;

FIG. **9A** illustrates a search screen in accordance with one or more aspects of the present disclosure;

FIG. **9B** illustrates a data summary dialog that enables a user to select various data sources in accordance with one or more aspects of the present disclosure;

FIG. **10A** illustrates a key indicators view in accordance with one or more aspects of the present disclosure;

FIG. **10B** illustrates an incident review dashboard in accordance with one or more aspects of the present disclosure;

FIG. **10C** illustrates a proactive monitoring tree in accordance with one or more aspects of the present disclosure;

FIG. **10D** illustrates a screen displaying both log data and performance data in accordance with one or more aspects of the present disclosure;

FIG. **11** depicts a block diagram of an example computing device operating in accordance with one or more aspects of the present disclosure.

DETAILED DESCRIPTION

Disclosed herein are systems and methods for assigning scores to objects based on evaluating triggering conditions applied to datasets produced by search queries.

An example data aggregation and analysis system may aggregate heterogeneous machine-generated data received from various sources, including servers, databases, applications, networks, etc. The aggregated source data may comprise a plurality of events. An event may be represented by a data structure that is associated with a certain point in time and comprises a portion of raw machine data (i.e., machine-generated data). The system may be configured to perform real-time indexing of the source data and to execute real-time, scheduled, or historic searches on the source data. A search query may comprise one or more search terms specifying the search criteria. Search terms may include keywords, phrases, Boolean expressions, regular expressions, field names, name-value pairs, etc. The search criteria may comprise a filter specifying relative or absolute time values, to limit the scope of the search by a specific time value or a specific time range.

The example data aggregation and analysis system executing a search query may evaluate the data relative to the search criteria to produce a resulting dataset. The resulting dataset may comprise one or more data items representing one or more portions of the source data that satisfy the search criteria. Alternatively, the resulting dataset may just include an indication that the search criteria have been satisfied. Yet alternatively, the resulting dataset may include a number indicating how many times the search criteria have been satisfied.

The example data aggregation and analysis system may be employed to assign scores to various objects associated with a distributed computer system (e.g., an enterprise system comprising a plurality of computer systems and peripheral devices interconnected by a plurality of networks). An object may represent such things as an entity (such as a particular user or a particular organization), or an asset (such as a particular computer system or a particular application). In various illustrative examples, the scores assigned by the data aggregation and analysis system may represent security risk scores, system performance scores (indicating the performance of components such as hosts, servers, routers, switches, attached storage, or virtual machines in an IT environment), or application performance scores. In certain implementations, the scores assigned by the data aggregation and analysis system may belong to a certain scale. Alterna-

tively, the scores may be represented by values which do not belong to any scale. In certain implementations, the scores may be represented by dimensionless values.

In certain implementations, the data aggregation and analysis system may adjust, by a certain score modifier value, a risk score assigned to a certain object responsive to determining that at least a portion of a dataset produced by executing a search query satisfies a certain triggering condition. A triggering condition can be any condition that is intended to trigger a specific action. An example triggering condition can trigger an action every time search criteria are satisfied (e.g., every time a specific user has a failed authentication attempt). Another example is a triggering condition that can trigger an action when a number specifying how many times search criteria have been satisfied exceeds a threshold (e.g., when the number of failed authentication logins of a specific user exceeds 5). Yet another example is a triggering condition that pertains to aggregating a dataset returned by the search query to form statistics pertaining to one or more attributes of the dataset that were used for aggregation, where the triggering condition can trigger an action when the aggregated statistics meet a criteria such as exceeding a threshold, being under a threshold, or falling within a specified range. For example, a dataset returned by the search query may include failed authentication attempts for logging into any application (e.g., email application, CRM application, HCM application, etc.) and initiated by numerous source IP (Internet Protocol) addresses; the dataset may be aggregated to produce counts of failed authentication attempts on a per application per source basis (i.e., first aggregated by application and then further aggregated by source); and the triggering condition may trigger an action when any of the counts exceeds a threshold. It should be noted that in some implementations, the evaluation of the aggregated statistics can be handled as part of the search query, and not as part of the triggering condition evaluation (where the triggering condition either triggers every time the search criteria are met or triggers when the search criteria are met at least a minimum number of times when the search is run).

A triggering condition may be applied to a dataset produced by a search query that is executed by the system either in real time or according to a certain schedule. Whenever at least a portion of the dataset returned by the search satisfies the triggering condition, a risk score associated with a certain object to which the portion of the dataset pertains (e.g., an object that is directly or indirectly referenced by the portion of the dataset) may be modified (increased or decreased) by a certain risk score modifier value.

In an illustrative example, the risk score associated with an object may be modified every time the dataset returned by the search query includes an indicator that the search criteria of the search query are satisfied. Alternatively, the risk score associated with an object may be modified when the number of times the search criteria are satisfied exceeds a threshold. Yet alternatively, the risk score associated with an object may be modified when the aggregated statistics pertaining to the dataset returned by the query meet specified criteria (such as exceeding a threshold, being under a threshold, or falling within a specified range).

The risk score modifier value may be determined based on values of one or more fields of the portion of the dataset that has triggered the risk score modification, as described in more detail below.

The data aggregation and analysis system may be further configured to present the assigned risk scores via a graphical user interface (GUI) of a client computing device (e.g., a

desktop computing device or a mobile computing device), as described in more detail below.

Accordingly, implementations of the present disclosure provide an effective mechanism for managing IT security, IT operations, and other aspects of the functioning of distributed computer or information technology systems by adjusting scores (e.g., security risk scores or performance scores) of objects in response to detecting an occurrence of certain conditions as indicated by data (e.g., the machine derived) produced by the system. The adjusted scores of objects are then visually presented to a user such as a system administrator to allow the user to quickly identify objects with respect to which certain remedial actions should be taken.

Various aspects of the methods and systems are described herein by way of example, rather than by way of limitation. The methods described herein may be implemented by hardware (e.g., general purpose and/or specialized processing devices, and/or other devices and associated circuitry), software (e.g., instructions executable by a processing device), or a combination thereof.

FIG. 1 schematically illustrates an example GUI for specifying security score modification rules, including search queries, triggering conditions, and other information to be utilized by the system for assigning and/or modifying security risk scores associated with various objects, in accordance with one or more aspects of the present disclosure. While FIG. 1 and the corresponding description illustrate and refer to security risk scores, same and/or similar GUI elements, systems and methods may be utilized by the example data aggregation and analysis system for specifying data searches, triggering conditions, and other information to be utilized by the system for assigning other types of scores, such as system performance scores or application performance scores. System or application performance scores may be utilized for quantifying various aspects of system or application performance, e.g., in situations when no single objectively measurable attribute or characteristic may reasonably be employed for the stated purpose.

As schematically illustrated by FIG. 1, example GUI 100 may comprise one or more input fields for specifying search identifiers such as an alphanumeric name 107 and an alphanumeric description 110 of the security score modification rule defined by the search. Example GUI 100 may further comprise a drop-down list for selecting the application context 115 associated with the search. In an illustrative example, the application context may identify an application of a certain platform, such as the SPLUNK® ENTERPRISE system produced by Splunk Inc. of San Francisco, Calif., which is described in more details herein below).

In certain implementations, example GUI 100 may further comprise a text box 120 for specifying a search query string comprising one or more search terms specifying the search criteria. The search query string may comply with the syntax of a certain query language supported by the data aggregation and retrieval system, such as Splunk Search Processing Language (SPL) which is further described herein below. Alternatively, the search query may be specified using other input mechanisms, such as selecting the search query from a list of pre-defined search queries, or building the search query using a wizard comprising a plurality of pre-defined input fields.

Example GUI 100 may further comprise a start time and end time input field's 125A-125B. In an illustrative example, the start time and end time may define a time window specified relative to the current time (e.g., from 5 minutes before the current time to the current time). The start time and end time input fields specify the time range limiting the scope of the search, i.e., instructing the example data aggregation and

analysis system to perform the search query on the source data items (e.g., events) that have timestamps falling within the specified time range.

Example GUI **100** may further comprise a schedule input field **130** to define the schedule according to which the search query should be executed by the example data aggregation and analysis system. The schedule may be represented by a data structure comprising values of one or more scheduling parameters (e.g., minute, hour, day, month, and/or day-of-week). Executing search query according to a certain schedule may be useful, e.g., for a search query that has its scope limited by a time window specified relative to the time the query query is run (e.g., from 5 minutes before the time of beginning execution of the query to the time of beginning execution of the query).

Example GUI **100** may further comprise a throttling window input field **135** and a grouping field selection field **140** to define a throttling condition. The throttling condition may be utilized to suppress, for a certain period of time (e.g., for a number of seconds specified by field **135**), triggering the score modification and/or other actions associated with the search query. Grouping field **140** may be utilized to select a field by the value of which the search results should be grouped for evaluating the throttling condition. In other words, the example data aggregation and analysis system may suppress the actions associated with the search query for a specified number of seconds for the search results that include the same value in the specified field (e.g., the same user identifier in the "user" field shown in the grouping field **140** in the illustrative example of FIG. **1**).

Example GUI **100** may further comprise a "Create risk score modifier" checkbox **145** specifying that the specified risk score modification actions should be performed based on a trigger condition resulting from execution of the search query.

As noted herein above, the data aggregation and analysis system may be configured to adjust, by a certain risk score modifier value, the risk score assigned to one or more objects responsive to determining that at least a portion of a dataset produced by the search satisfies a particular triggering condition. In an illustrative example, the risk score associated with an object may be modified every time the search query returns an indicator that the search criteria are satisfied. Alternatively, the risk score associated with an object may be modified when the number of times the search criteria were satisfied exceeds a threshold. In yet another example, the risk score associated with an object may be modified when the aggregated statistics pertaining to the dataset returned by the search query meets certain criteria (e.g., exceeding a threshold, being under a threshold, or falling within a certain range).

In the illustrative example of FIG. **1**, the risk score modifier value is specified by input field **150** as a constant integer value. Alternatively, the risk score modifier value may be determined by performing certain calculations on one or more data items (referenced by the corresponding fields names) that are identified by the search query as meeting the critieria of the query. Risk score modifiers may be provided by positive or negative values. A positive risk score modifier value may indicate that the total risk score associated with an object should be increased (e.g., if the object represents a user who has been engaged in an activity associated with an elevated risk score value). A negative risk score modifier value may indicate that the total risk score associated with an object should be decreased (e.g., if the object represents a system administrator who has been engaged in an activity that, if performed by a non-privileged user, would appear as associated with an elevated risk score value). The object

whose score should be modified may be identified by a field in the data meeting the search criteria and/or triggering condition.

In an illustrative example, each occurrence of a certain pre-defined state or situation defined by the search criteria may necessitate modifying a risk score assigned to an object by a certain integer value. The arithmetic expression defining the risk score modifier may specify that the integer value should be multiplied by the number of occurrences of the state or situation returned by the search query (e.g., if a failed login attempt increases a user's risk score by 10, the arithmetic expression defining the risk score modifier may specify the value being equal to 10*N, wherein N is the number of failed login attempts). In another illustrative example, the risk score modifier may be proportional to a metric associated with a certain activity (e.g., if each kilobyte of VPN traffic increases the user's risk score by 12, the arithmetic expression defining the risk score modifier may specify the value being equal to 12*T/1024, wherein T is the amount of VPN traffic, in bytes, associated with the user, and 1024 is the number of bytes in a kilobyte; in this case, the number of kilobytes of VPN traffic may be extracted from a field in the data that met the search criteria and resulted in the triggering condition). Likewise, the object whose score should be modified may be identified from a field in the data that met the search criteria and resulted in the triggering condition.

Example GUI **100** may further comprise a risk object field **155** to identify the object whose risk score should be modified by the example data aggregation and analysis system. The risk object may be identified by a data item (such as by a field in the data item that is referenced by the field name **155**) included in a dataset produced by the search query. Example objects may include a user, a computer system, a network, an application, etc.

In certain implementations, should the identified field name contain an empty value, the example data aggregation and analysis system may apply the risk score modifier to the risk score associated with a placeholder (or fictitious) object used for accumulating risk score modifiers that cannot be traced to a particular known object. In an illustrative example, the fictitious object to which risk score modifiers associated with unidentified objects are applied may be referenced by a symbolic name (e.g., UNKNOWN object). Applying risk score modifiers associated with unidentified objects to a fictitious object may be utilized to attract a user's attention to the fact that certain objects associated with non-zero (or even significant) risk scores could not be identified by the system.

Example GUI **100** may further comprise a risk object type field **160** to identify the type of risk object **155**. In various illustrative examples, the risk object type may be represented by one of the following types: an entity (such as a user or an organization), an asset (such as a computer system or an application), or a user-defined type (e.g., a building).

Example GUI **100** may further comprise one or more action check-boxes **165A-165C** to specify one or more actions to be performed by the system responsive to determining that at least a portion of the dataset produced by executing the specified search query satisfies the specified triggering condition. The actions may include, for example, sending an e-mail message comprising the risk score modifier value and/or at least part of the dataset that has triggered the risk score modification, creating an RSS feed comprising the risk score modifier value and/or at least part of the dataset that has triggered the risk score modification, and/or executing a shell script having at least one parameter defined based on the score.

In certain implementations, the specified actions may be performed with respect to each result produced by the search query defined by query input field **110** (in other words, the simplest triggering condition is applied to the resulting dataset requiring that the resulting dataset comprise a non-zero number of results). Alternatively, an additional triggering condition may be applied to the resulting dataset produced by the search query (e.g., comparing the number of data items in the resulting dataset produced to a certain configurable integer value or performing a secondary search on the dataset produced by executing the search query).

In certain implementations, responsive to modifying a score assigned to the primary object, the example data aggregation and analysis system may also modify scores assigned to one or more additional objects that are associated with the primary object. For example, if a security risk score assigned to an object representing a user's laptop is modified responsive to a certain triggering condition, the system may further modify the security risk score assigned to the object representing the user himself. In an illustrative example, the example data aggregation and analysis system may identify one or more additional objects associated with the primary objects based on one or more object association rules. In another illustrative example, the example data aggregation and analysis system may identify one or more additional objects associated with the primary objects based on performing a secondary search using a pre-defined or dynamically constructed search query. The risk score modifier value to be applied to the associated additional object may be determined based on the risk score modifier value of the primary object and/or one or more object association rules. In an illustrative example, an object association rule may specify that the risk score modifier value of an additional object (e.g., a user) associated with a primary object (e.g., the user's laptop) may be determined as a certain fraction of the risk score modifier value of the primary object.

As noted herein above, the example data aggregation and analysis system may be further configured to present the assigned security risk scores via a graphical user interface (GUI) of a client computing device (e.g., a desktop computing device or a mobile computing device). FIG. **2** schematically illustrates an example GUI for visually presenting security risk scores assigned to a plurality of objects, in accordance with one or more aspects of the present disclosure. While FIG. **2** and the corresponding description illustrate and refer to security risk scores, the same and/or similar GUI elements, systems, and methods may be utilized by the example data aggregation and analysis system for visually presenting other types of scores, such as system performance scores or application performance scores.

As schematically illustrated by FIG. **2**, example GUI **200** may comprise several panels **210A-210N** to dynamically present graphical and/or textual information associated with security risk scores. In the illustrative example of FIG. **2**, example GUI **200** may further comprise a panel **210A** showing a graph **232** representing the total risk score value assigned to a selected set of objects within the time period identified by time period selection dropdown control **234**. The set of objects for displaying the risk score values may be specified by the risk object identifier (input field **236**), and/or risk object type (input field **238**). The risk score values may be further filtered by specifying the risk object sources (e.g., risk score modification rules) via input field **240**.

Example GUI **200** may further comprise panel **210B** representing, in a rectangular table, risk scores (column **242**) assigned to a plurality of objects identified by symbolic names (column **244**). The set of objects for which the scores are displayed and/or the risk scores to be displayed may be limited by one or more parameters specified by one or more fields of the input panel **210A**, such as only displaying risk modifiers resulting from selected search/trigger combinations (source pull down menu **240**), only displaying objects of a given object type (pull down menu **238**), only displaying particular objects entered in the box **236**, or calculating the scores for displayed objects by aggregating only those risk score modifiers for each displayed object that occur with a time range specified in time-range pulldown menu **234**.

The table entries displayed within display panel **210B** may be sorted, e.g., in a descending order of total risk score associated with the corresponding object, thus allowing the user to focus on the objects associated with the largest values of risk security scores. Panel **210B** may further comprise column **246** showing the object type (e.g., a user type, a system type, or a user-defined type). In the illustrative example of FIG. **2**, the object types shown in column **246** may match the object type specified by pull-down menu **238**. Panel **210A** may further comprise column **248** showing the number of search/trigger/score rules (each of which is referred to as a "source") contributing to the total risk score associated with the object identified by column **244** (or, in other words, the number of rules for which the object has satisfied the triggering condition). Panel **210A** may further comprise column **250** showing the number of individual risk score modifiers reflected by the total risk score associated with the object identified by column **242** (or, in other words, the number of times when a triggering condition was met by the object).

Example GUI **200** may further comprise panel **210C** representing, in a rectangular table, aggregate risk score values of the various risk modifiers grouped by the sources (e.g., risk score modification rules identified by symbolic names in column **212**) that generated the risk modifiers and ordered in the descending order of the risk score value (column **214**). Panel **210C** may further comprise column **216** showing the number of objects having their risk score values modified by the corresponding source, and column **218** showing the number of individual risk score modifiers reflected by the total risk score value identified by column **214**.

Example GUI **200** may further comprise a panel **210N** representing, in a rectangular table, the most recently created risk modifiers (the score for which is provided in column **220**, and a description of the risk score rule that generated the risk modifier is provided in column **230**). Each row may display the object whose score is affected by the risk modifier represented by that row (column **222**). The table entries may be ordered in the reverse time order (most recent entries first) based on the risk modifier creation time (column **224**). Panel **210N** may further comprise column **226** showing the object type for the object in column **222**, column **228** showing the risk modifier source (e.g., a symbolic name referencing the risk score modification rule that generated the risk modifier represented in a given row).

In certain implementations, the example data aggregation and analysis system may allow a user to "drill down" to the underlying data that has triggered a particular risk score modifier. For example, responsive to receiving the user's selection of a particular risk score modifier, the system may display further information pertaining to the selected modifier, such as the underlying portion of the data that has triggered the risk score modifier.

In certain implementations, the example data aggregation and analysis system may provide an "ad hoc" score modification interface to allow a user to adjust risk score modifiers assigned to certain objects. In an illustrative example, a user

may increase or decrease a risk score value assigned to a certain object or a group of objects.

FIGS. 3A-3C depict flow diagrams of example methods 300A-300B for assigning scores to objects based on evaluating triggering conditions applied to datasets produced by search queries. Methods 300A-300B and/or each of their respective individual functions, routines, subroutines, or operations may be performed by one or more general purpose and/or specialized processing devices. Two or more functions, routines, subroutines, or operations of methods 300A-300B may be performed in parallel or in an order that may differ from the order described above. In certain implementations, one or more of methods 300A-300B may be performed by a single processing thread. Alternatively, methods 300A-300B may be performed by two or more processing threads, each thread executing one or more individual functions, routines, subroutines, or operations of the respective method. In an illustrative example, the processing threads implementing methods 300A-300B may be synchronized (e.g., using semaphores, critical sections, and/or other thread synchronization mechanisms). Alternatively, the processing threads implementing methods 300A-300B may be executed asynchronously with respect to each other. In an illustrative example, methods 300A-300B may be performed by an example computing device 1000 described herein below with references to FIG. 11. In another illustrative example, methods 300A-300B may be performed by a distributed computer system comprising two or more example computing devices 1000.

FIG. 3A depict a flow diagram of an example method 300A for modifying score values assigned to certain objects based on search query results, in accordance with one or more aspects of the present disclosure.

Referring to FIG. 3A, at block 310, the computer system implementing the method may execute a search query. In an illustrative example, the search query may represent a real-time search (e.g., may repeatedly be executed by a certain process or thread in an indefinite loop which may be interrupted by occurrences of certain terminating conditions). In another illustrative example, the search query may represent a scheduled search (e.g., may be executed according to a certain schedule), as described in more details herein above.

Responsive to determining, at block 315, that a portion of the dataset produced by the search query satisfies a triggering condition defined by a risk score modification rule associated with the search query, the processing may continue at block 320; otherwise, the processing associated with the current search query instance may terminate.

At block 320, the computer system may modify a risk score value of a certain primary object by a risk score modifier value. The primary object may be identified based on values of one or more fields of the portion of the dataset returned by the search query, in accordance with the risk score modification rule associated with the search query, as described in more details herein above. The risk score modifier values may be determined in accordance with the risk score modification rule associated with the search query. In an illustrative example, the risk score modifier value applicable to a certain object may be defined as a constant integer value. Alternatively, the risk score modifier value may be determined by performing certain calculations on one or more data items (e.g., by extracting values for fields in the data items that are used in the calculation) included in the resulting dataset produced by the search query. In an illustrative example, the risk score modifier value may be specified by a certain arithmetic expression. The arithmetic expression may comprise one or more arithmetic operations to be performed on two or more

operands. Each of the operands may be represented by a value of a data item (referenced by the corresponding field name) included in the resulting dataset produced by the search query or by a certain constant value.

At block 330, the computer system may modify risk score values of certain objects associated with the primary object. The example data aggregation and analysis system may identify one or more objects associated with the primary object based on one or more object association rules. The risk score modifier value to be applied to the associated additional object may be determined based on the risk score modifier value of the primary object and/or one or more object association rules, as described in more details herein above with references to FIG. 1.

FIG. 3B depicts a flow diagram of an example method 300B for presenting score modifier information, in accordance with one or more aspects of the present disclosure. As noted herein above, method 300B may be implemented by a server (e.g., a presentation server) and/or by one or more clients of the distributed computer system operating in accordance with one or more aspects of the present disclosure.

Referring to FIG. 3B, at block 350, the computer system implementing the method may sort the score modifier information associated with certain objects in an order reflecting the corresponding score modifier values (e.g., in the descending order of the score modifier values). The objects for displaying the associated score modifier information may be selected by a user via a GUI, as described in more details herein above with reference to FIG. 2.

At block 355, the computer system may cause the score modifier information to be displayed by a client computing device, as described in more details herein above with reference to FIG. 2.

Responsive to receiving, at block 360, a user's selection of a particular score modifier of the displayed score modifiers, the computer system may, at block 365, cause further information pertaining to the selected modifier to be displayed, including the underlying portion of the dataset that has triggered the risk score modifier.

The systems and methods described herein above may be employed by various data processing systems, e.g., data aggregation and analysis systems. In certain implementations, the example data aggregation and analysis system may perform search queries on data (e.g., relating to the security of an IT environment or related to the performance of components in that IT environment) that is stored as "events," wherein each event comprises a portion of machine data generated by the computer or IT environment and that is correlated with a specific point in time. In various illustrative examples, the data processing system may be represented by the SPLUNK® ENTERPRISE system produced by Splunk Inc. of San Francisco, Calif., to store and process performance data. The data processing system may be configured to execute search queries as correlational searches, as described in more details herein below. In certain implementations, the risk scoring framework may be included in an application like the SPLUNK® APP FOR ENTERPRISE SECURITY.

Modern data centers often comprise thousands of host computer systems that operate collectively to service requests from even larger numbers of remote clients. During operation, these data centers generate significant volumes of performance data and diagnostic information that can be analyzed to quickly diagnose performance problems. In order to reduce the size of this performance data, the data is typically pre-processed prior to being stored based on anticipated data-analysis needs. For example, pre-specified data items can be extracted from the performance data and stored in a database

to facilitate efficient retrieval and analysis at search time. However, the rest of the performance data is not saved and is essentially discarded during pre-processing. As storage capacity becomes progressively cheaper and more plentiful, there are fewer incentives to discard this performance data and many reasons to keep it.

This plentiful storage capacity is presently making it feasible to store massive quantities of minimally processed performance data at "ingestion time" for later retrieval and analysis at "search time." Note that performing the analysis operations at search time provides greater flexibility because it enables an analyst to search all of the performance data, instead of searching pre-specified data items that were stored at ingestion time. This enables the analyst to investigate different aspects of the performance data instead of being confined to the pre-specified set of data items that were selected at ingestion time.

However, analyzing massive quantities of heterogeneous performance data at search time can be a challenging task. A data center may generate heterogeneous performance data from thousands of different components, which can collectively generate tremendous volumes of performance data that can be time-consuming to analyze. For example, this performance data can include data from system logs, network packet data, sensor data, and data generated by various applications. Also, the unstructured nature of much of this performance data can pose additional challenges because of the difficulty of applying semantic meaning to unstructured data, and the difficulty of indexing and querying unstructured data using traditional database systems.

These challenges can be addressed by using an event-based system, such as the SPLUNK® ENTERPRISE system produced by Splunk Inc. of San Francisco, Calif., to store and process performance data. The SPLUNK® ENTERPRISE system is the leading platform for providing real-time operational intelligence that enables organizations to collect, index, and harness machine-generated data from various websites, applications, servers, networks, and mobile devices that power their businesses. The SPLUNK® ENTERPRISE system is particularly useful for analyzing unstructured performance data, which is commonly found in system log files. Although many of the techniques described herein are explained with reference to the SPLUNK® ENTERPRISE system, the techniques are also applicable to other types of data server systems.

In the SPLUNK® ENTERPRISE system, performance data is stored as "events," wherein each event comprises a collection of performance data and/or diagnostic information that is generated by a computer system and is correlated with a specific point in time. Events can be derived from "time series data," wherein time series data comprises a sequence of data points (e.g., performance measurements from a computer system) that are associated with successive points in time and are typically spaced at uniform time intervals. Events can also be derived from "structured" or "unstructured" data. Structured data has a predefined format, wherein specific data items with specific data formats reside at predefined locations in the data. For example, structured data can include data items stored in fields in a database table. In contrast, unstructured data does not have a predefined format. This means that unstructured data can comprise various data items having different data types that can reside at different locations. For example, when the data source is an operating system log, an event can include one or more lines from the operating system log containing raw data that includes different types of performance and diagnostic information associated with a specific point in time. Examples of data sources

from which an event may be derived include, but are not limited to: web servers; application servers; databases; firewalls; routers; operating systems; and software applications that execute on computer systems, mobile devices, and sensors. The data generated by such data sources can be produced in various forms including, for example and without limitation, server log files, activity log files, configuration files, messages, network packet data, performance measurements and sensor measurements. An event typically includes a timestamp that may be derived from the raw data in the event, or may be determined through interpolation between temporally proximate events having known timestamps.

The SPLUNK® ENTERPRISE system also facilitates using a flexible schema to specify how to extract information from the event data, wherein the flexible schema may be developed and redefined as needed. Note that a flexible schema may be applied to event data "on the fly," when it is needed (e.g., at search time), rather than at ingestion time of the data as in traditional database systems. Because the schema is not applied to event data until it is needed (e.g., at search time), it is referred to as a "late-binding schema."

During operation, the SPLUNK® ENTERPRISE system starts with raw data, which can include unstructured data, machine data, performance measurements or other time-series data, such as data obtained from weblogs, syslogs, or sensor readings. It divides this raw data into "portions," and optionally transforms the data to produce timestamped events. The system stores the timestamped events in a data store, and enables a user to run queries against the data store to retrieve events that meet specified criteria, such as containing certain keywords or having specific values in defined fields. Note that the term "field" refers to a location in the event data containing a value for a specific data item.

As noted above, the SPLUNK® ENTERPRISE system facilitates using a late-binding schema while performing queries on events. A late-binding schema specifies "extraction rules" that are applied to data in the events to extract values for specific fields. More specifically, the extraction rules for a field can include one or more instructions that specify how to extract a value for the field from the event data. An extraction rule can generally include any type of instruction for extracting values from data in events. In some cases, an extraction rule comprises a regular expression, in which case the rule is referred to as a "regex rule."

In contrast to a conventional schema for a database system, a late-binding schema is not defined at data ingestion time. Instead, the late-binding schema can be developed on an ongoing basis until the time a query is actually executed. This means that extraction rules for the fields in a query may be provided in the query itself, or may be located during execution of the query. Hence, as an analyst learns more about the data in the events, the analyst can continue to refine the late-binding schema by adding new fields, deleting fields, or changing the field extraction rules until the next time the schema is used by a query. Because the SPLUNK® ENTERPRISE system maintains the underlying raw data and provides a late-binding schema for searching the raw data, it enables an analyst to investigate questions that arise as the analyst learns more about the events.

In the SPLUNK® ENTERPRISE system, a field extractor may be configured to automatically generate extraction rules for certain fields in the events when the events are being created, indexed, or stored, or possibly at a later time. Alternatively, a user may manually define extraction rules for fields using a variety of techniques.

Also, a number of "default fields" that specify metadata about the events rather than data in the events themselves can

13

be created automatically. For example, such default fields can specify: a timestamp for the event data; a host from which the event data originated; a source of the event data; and a source type for the event data. These default fields may be determined automatically when the events are created, indexed or stored.

In some embodiments, a common field name may be used to reference two or more fields containing equivalent data items, even though the fields may be associated with different types of events that possibly have different data formats and different extraction rules. By enabling a common field name to be used to identify equivalent fields from different types of events generated by different data sources, the system facilitates use of a "common information model" (CIM) across the different data sources.

FIG. 4 presents a block diagram of an exemplary event-processing system 100, similar to the SPLUNK® ENTERPRISE system. System 100 includes one or more forwarders 101 that collect data obtained from a variety of different data sources 105, and one or more indexers 102 that store, process, and/or perform operations on this data, wherein each indexer operates on data contained in a specific data store 103. These forwarders and indexers can comprise separate computer systems in a data center, or may alternatively comprise separate processes executing on various computer systems in a data center.

During operation, the forwarders 101 identify which indexers 102 will receive the collected data and then forward the data to the identified indexers. Forwarders 101 can also perform operations to strip out extraneous data and detect timestamps in the data. The forwarders next determine which indexers 102 will receive each data item and then forward the data items to the determined indexers 102.

Note that distributing data across different indexers facilitates parallel processing. This parallel processing can take place at data ingestion time, because multiple indexers can process the incoming data in parallel. The parallel processing can also take place at search time, because multiple indexers can search through the data in parallel.

System 100 and the processes described below with respect to FIGS. 5-10 are further described in "Exploring Splunk Search Processing Language (SPL) Primer and Cookbook" by David Carasso, CITO Research, 2012, and in "Optimizing Data Analysis With a Semi-Structured Time Series Database" by Ledion Bitincka, Archana Ganapathi, Stephen Sorkin, and Steve Zhang, SLAML, 2010, each of which is hereby incorporated herein by reference in its entirety for all purposes.

FIG. 5 presents a flowchart illustrating how an indexer processes, indexes, and stores data received from forwarders in accordance with the disclosed embodiments. At block 201, the indexer receives the data from the forwarder. Next, at block 202, the indexer apportions the data into events. Note that the data can include lines of text that are separated by carriage returns or line breaks and an event may include one or more of these lines. During the apportioning process, the indexer can use heuristic rules to automatically determine the boundaries of the events, which for example coincide with line boundaries. These heuristic rules may be determined based on the source of the data, wherein the indexer can be explicitly informed about the source of the data or can infer the source of the data by examining the data. These heuristic rules can include regular expression-based rules or delimiter-based rules for determining event boundaries, wherein the event boundaries may be indicated by predefined characters or character strings. These predefined characters may include punctuation marks or other special characters including, for example, carriage returns, tabs, spaces or line breaks. In some

14

cases, a user can fine-tune or configure the rules that the indexers use to determine event boundaries in order to adapt the rules to the user's specific requirements.

Next, the indexer determines a timestamp for each event at block 203. As mentioned above, these timestamps can be determined by extracting the time directly from data in the event, or by interpolating the time based on timestamps from temporally proximate events. In some cases, a timestamp can be determined based on the time the data was received or generated. The indexer subsequently associates the determined timestamp with each event at block 204, for example by storing the timestamp as metadata for each event.

Then, the system can apply transformations to data to be included in events at block 205. For log data, such transformations can include removing a portion of an event (e.g., a portion used to define event boundaries, extraneous text, characters, etc.) or removing redundant portions of an event. Note that a user can specify portions to be removed using a regular expression or any other possible technique.

Next, a keyword index can optionally be generated to facilitate fast keyword searching for events. To build a keyword index, the indexer first identifies a set of keywords in block 206. Then, at block 207 the indexer includes the identified keywords in an index, which associates each stored keyword with references to events containing that keyword (or to locations within events where that keyword is located). When an indexer subsequently receives a keyword-based query, the indexer can access the keyword index to quickly identify events containing the keyword.

In some embodiments, the keyword index may include entries for name-value pairs found in events, wherein a name-value pair can include a pair of keywords connected by a symbol, such as an equals sign or colon. In this way, events containing these name-value pairs can be quickly located. In some embodiments, fields can automatically be generated for some or all of the name-value pairs at the time of indexing. For example, if the string "dest=10.0.1.2" is found in an event, a field named "dest" may be created for the event, and assigned a value of "10.0.1.2."

Finally, the indexer stores the events in a data store at block 208, wherein a timestamp can be stored with each event to facilitate searching for events based on a time range. In some cases, the stored events are organized into a plurality of buckets, wherein each bucket stores events associated with a specific time range. This not only improves time-based searches, but it also allows events with recent timestamps that may have a higher likelihood of being accessed to be stored in faster memory to facilitate faster retrieval. For example, a bucket containing the most recent events can be stored as flash memory instead of on hard disk.

Each indexer 102 is responsible for storing and searching a subset of the events contained in a corresponding data store 103. By distributing events among the indexers and data stores, the indexers can analyze events for a query in parallel, for example using map-reduce techniques, wherein each indexer returns partial responses for a subset of events to a search head that combines the results to produce an answer for the query. By storing events in buckets for specific time ranges, an indexer may further optimize searching by looking only in buckets for time ranges that are relevant to a query.

Moreover, events and buckets can also be replicated across different indexers and data stores to facilitate high availability and disaster recovery as is described in U.S. patent application Ser. No. 14/266,812 filed on 30 Apr. 2014, and in U.S. application patent Ser. No. 14/266,817 also filed on 30 Apr. 2014.

FIG. 6 presents a flowchart illustrating how a search head and indexers perform a search query in accordance with the disclosed embodiments. At the start of this process, a search head receives a search query from a client at block 301. Next, at block 302, the search head analyzes the search query to determine what portions can be delegated to indexers and what portions need to be executed locally by the search head. At block 303, the search head distributes the determined portions of the query to the indexers. Note that commands that operate on single events can be trivially delegated to the indexers, while commands that involve events from multiple indexers are harder to delegate.

Then, at block 304, the indexers to which the query was distributed search their data stores for events that are responsive to the query. To determine which events are responsive to the query, the indexer searches for events that match the criteria specified in the query. This criteria can include matching keywords or specific values for certain fields. In a query that uses a late-binding schema, the searching operations in block 304 may involve using the late-binding scheme to extract values for specified fields from events at the time the query is processed. Next, the indexers can either send the relevant events back to the search head, or use the events to calculate a partial result, and send the partial result back to the search head.

Finally, at block 305, the search head combines the partial results and/or events received from the indexers to produce a final result for the query. This final result can comprise different types of data depending upon what the query is asking for. For example, the final results can include a listing of matching events returned by the query, or some type of visualization of data from the returned events. In another example, the final result can include one or more calculated values derived from the matching events.

Moreover, the results generated by system 100 can be returned to a client using different techniques. For example, one technique streams results back to a client in real-time as they are identified. Another technique waits to report results to the client until a complete set of results is ready to return to the client. Yet another technique streams interim results back to the client in real-time until a complete set of results is ready, and then returns the complete set of results to the client. In another technique, certain results are stored as "search jobs," and the client may subsequently retrieve the results by referencing the search jobs.

The search head can also perform various operations to make the search more efficient. For example, before the search head starts executing a query, the search head can determine a time range for the query and a set of common keywords that all matching events must include. Next, the search head can use these parameters to query the indexers to obtain a superset of the eventual results. Then, during a filtering stage, the search head can perform field-extraction operations on the superset to produce a reduced set of search results.

FIG. 7 presents a block diagram illustrating how fields can be extracted during query processing in accordance with the disclosed embodiments. At the start of this process, a search query 402 is received at a query processor 404. Query processor 404 includes various mechanisms for processing a query, wherein these mechanisms can reside in a search head 104 and/or an indexer 102. Note that the exemplary search query 402 illustrated in FIG. 7 is expressed in Search Processing Language (SPL), which is used in conjunction with the SPLUNK® ENTERPRISE system. SPL is a pipelined search language in which a set of inputs is operated on by a first command in a command line, and then a subsequent

command following the pipe symbol "I" operates on the results produced by the first command, and so on for additional commands. Search query 402 can also be expressed in other query languages, such as the Structured Query Language ("SQL") or any suitable query language.

Upon receiving search query 402, query processor 404 sees that search query 402 includes two fields "IP" and "target." Query processor 404 also determines that the values for the "IP" and "target" fields have not already been extracted from events in data store 414, and consequently determines that query processor 404 needs to use extraction rules to extract values for the fields. Hence, query processor 404 performs a lookup for the extraction rules in a rule base 406, wherein rule base 406 maps field names to corresponding extraction rules and obtains extraction rules 408-409, wherein extraction rule 408 specifies how to extract a value for the "IP" field from an event, and extraction rule 409 specifies how to extract a value for the "target" field from an event. As is illustrated in FIG. 7, extraction rules 408-409 can comprise regular expressions that specify how to extract values for the relevant fields. Such regular-expression-based extraction rules are also referred to as "regex rules." In addition to specifying how to extract field values, the extraction rules may also include instructions for deriving a field value by performing a function on a character string or value retrieved by the extraction rule. For example, a transformation rule may truncate a character string, or convert the character string into a different data format. In some cases, the query itself can specify one or more extraction rules.

Next, query processor 404 sends extraction rules 408-409 to a field extractor 412, which applies extraction rules 408-409 to events 416-418 in a data store 414. Note that data store 414 can include one or more data stores, and extraction rules 408-409 can be applied to large numbers of events in data store 414, and are not meant to be limited to the three events 416-418 illustrated in FIG. 7. Moreover, the query processor 404 can instruct field extractor 412 to apply the extraction rules to all the events in a data store 414, or to a subset of the events that have been filtered based on some criteria.

Next, field extractor 412 applies extraction rule 408 for the first command "Search IP="10*" to events in data store 414 including events 416-418. Extraction rule 408 is used to extract values for the IP address field from events in data store 414 by looking for a pattern of one or more digits, followed by a period, followed again by one or more digits, followed by another period, followed again by one or more digits, followed by another period, and followed again by one or more digits. Next, field extractor 412 returns field values 420 to query processor 404, which uses the criterion IP="10*" to look for IP addresses that start with "10". Note that events 416 and 417 match this criterion, but event 418 does not, so the result set for the first command is events 416-417.

Query processor 404 then sends events 416-417 to the next command "stats count target." To process this command, query processor 404 causes field extractor 412 to apply extraction rule 409 to events 416-417. Extraction rule 409 is used to extract values for the target field for events 416-417 by skipping the first four commas in events 416-417, and then extracting all of the following characters until a comma or period is reached. Next, field extractor 412 returns field values 421 to query processor 404, which executes the command "stats count target" to count the number of unique values contained in the target fields, which in this example produces the value "2" that is returned as a final result 422 for the query.

Note that query results can be returned to a client, a search head, or any other system component for further processing. In general, query results may include: a set of one or more

US 9,251,221 B1

17

events; a set of one or more values obtained from the events; a subset of the values; statistics calculated based on the values; a report containing the values; or a visualization, such as a graph or chart, generated from the values.

FIG. 9A illustrates an exemplary search screen 600 in accordance with the disclosed embodiments. Search screen 600 includes a search bar 602 that accepts user input in the form of a search string. It also includes a time range picker 612 that enables the user to specify a time range for the search. For "historical searches" the user can select a specific time range, or alternatively a relative time range, such as "today," "yesterday" or "last week." For "real-time searches," the user can select the size of a preceding time window to search for real-time events. Search screen 600 also initially displays a "data summary" dialog as is illustrated in FIG. 9B that enables the user to select different sources for the event data, for example by selecting specific hosts and log files.

After the search is executed, the search screen 600 can display the results through search results tabs 604, wherein search results tabs 604 includes: an "events tab" that displays various information about events returned by the search; a "statistics tab" that displays statistics about the search results; and a "visualization tab" that displays various visualizations of the search results. The events tab illustrated in FIG. 9A displays a timeline graph 605 that graphically illustrates the number of events that occurred in one-hour intervals over the selected time range. It also displays an events list 608 that enables a user to view the raw data in each of the returned events. It additionally displays a fields sidebar 606 that includes statistics about occurrences of specific fields in the returned events, including "selected fields" that are pre-selected by the user, and "interesting fields" that are automatically selected by the system based on pre-specified criteria.

The above-described system provides significant flexibility by enabling a user to analyze massive quantities of minimally processed performance data "on the fly" at search time instead of storing pre-specified portions of the performance data in a database at ingestion time. This flexibility enables a user to see correlations in the performance data and perform subsequent queries to examine interesting aspects of the performance data that may not have been apparent at ingestion time.

However, performing extraction and analysis operations at search time can involve a large amount of data and require a large number of computational operations, which can cause considerable delays while processing the queries. Fortunately, a number of acceleration techniques have been developed to speed up analysis operations performed at search time. These techniques include: (1) performing search operations in parallel by formulating a search as a map-reduce computation; (2) using a keyword index; (3) using a high performance analytics store; and (4) accelerating the process of generating reports. These techniques are described in more detail below.

To facilitate faster query processing, a query can be structured as a map-reduce computation, wherein the "map" operations are delegated to the indexers, while the corresponding "reduce" operations are performed locally at the search head. For example, FIG. 8 illustrates how a search query 501 received from a client at search head 104 can split into two phases, including: (1) a "map phase" comprising subtasks 502 (e.g., data retrieval or simple filtering) that may be performed in parallel and are "mapped" to indexers 102 for execution, and (2) a "reduce phase" comprising a merging operation 503 to be executed by the search head when the results are ultimately collected from the indexers.

18

During operation, upon receiving search query 501, search head 104 modifies search query 501 by substituting "stats" with "prestats" to produce search query 502, and then distributes search query 502 to one or more distributed indexers, which are also referred to as "search peers." Note that search queries may generally specify search criteria or operations to be performed on events that meet the search criteria. Search queries may also specify field names, as well as search criteria for the values in the fields or operations to be performed on the values in the fields. Moreover, the search head may distribute the full search query to the search peers as is illustrated in FIG. 6, or may alternatively distribute a modified version (e.g., a more restricted version) of the search query to the search peers. In this example, the indexers are responsible for producing the results and sending them to the search head. After the indexers return the results to the search head, the search head performs the merging operations 503 on the results. Note that by executing the computation in this way, the system effectively distributes the computational operations while minimizing data transfers.

As described above with reference to the flow charts in FIGS. 6 and 7, event-processing system 100 can construct and maintain one or more keyword indices to facilitate rapidly identifying events containing specific keywords. This can greatly speed up the processing of queries involving specific keywords. As mentioned above, to build a keyword index, an indexer first identifies a set of keywords. Then, the indexer includes the identified keywords in an index, which associates each stored keyword with references to events containing that keyword, or to locations within events where that keyword is located. When an indexer subsequently receives a keyword-based query, the indexer can access the keyword index to quickly identify events containing the keyword.

To speed up certain types of queries, some embodiments of system 100 make use of a high performance analytics store, which is referred to as a "summarization table," that contains entries for specific field-value pairs. Each of these entries keeps track of instances of a specific value in a specific field in the event data and includes references to events containing the specific value in the specific field. For example, an exemplary entry in a summarization table can keep track of occurrences of the value "94107" in a "ZIP code" field of a set of events, wherein the entry includes references to all of the events that contain the value "94107" in the ZIP code field. This enables the system to quickly process queries that seek to determine how many events have a particular value for a particular field, because the system can examine the entry in the summarization table to count instances of the specific value in the field without having to go through the individual events or do extractions at search time. Also, if the system needs to process all events that have a specific field-value combination, the system can use the references in the summarization table entry to directly access the events to extract further information without having to search all of the events to find the specific field-value combination at search time.

In some embodiments, the system maintains a separate summarization table for each of the above-described time-specific buckets that stores events for a specific time range, wherein a bucket-specific summarization table includes entries for specific field-value combinations that occur in events in the specific bucket. Alternatively, the system can maintain a separate summarization table for each indexer, wherein the indexer-specific summarization table only includes entries for the events in a data store that is managed by the specific indexer.

The summarization table can be populated by running a "collection query" that scans a set of events to find instances

of a specific field-value combination, or alternatively instances of all field-value combinations for a specific field. A collection query can be initiated by a user, or can be scheduled to occur automatically at specific time intervals. A collection query can also be automatically launched in response to a query that asks for a specific field-value combination.

In some cases, the summarization tables may not cover all of the events that are relevant to a query. In this case, the system can use the summarization tables to obtain partial results for the events that are covered by summarization tables, but may also have to search through other events that are not covered by the summarization tables to produce additional results. These additional results can then be combined with the partial results to produce a final set of results for the query. This summarization table and associated techniques are described in more detail in U.S. Pat. No. 8,682,925, issued on Mar. 25, 2014.

In some embodiments, a data server system such as the SPLUNK® ENTERPRISE system can accelerate the process of periodically generating updated reports based on query results. To accelerate this process, a summarization engine automatically examines the query to determine whether generation of updated reports can be accelerated by creating intermediate summaries. (This is possible if results from preceding time periods can be computed separately and combined to generate an updated report. In some cases, it is not possible to combine such incremental results, for example where a value in the report depends on relationships between events from different time periods.) If reports can be accelerated, the summarization engine periodically generates a summary covering data obtained during a latest non-overlapping time period. For example, where the query seeks events meeting a specified criteria, a summary for the time period includes only events within the time period that meet the specified criteria. Similarly, if the query seeks statistics calculated from the events, such as the number of events that match the specified criteria, then the summary for the time period includes the number of events in the period that match the specified criteria.

In parallel with the creation of the summaries, the summarization engine schedules the periodic updating of the report associated with the query. During each scheduled report update, the query engine determines whether intermediate summaries have been generated covering portions of the time period covered by the report update. If so, then the report is generated based on the information contained in the summaries. Also, if additional event data has been received and has not yet been summarized, and is required to generate the complete report, the query can be run on this additional event data. Then, the results returned by this query on the additional event data, along with the partial results obtained from the intermediate summaries, can be combined to generate the updated report. This process is repeated each time the report is updated. Alternatively, if the system stores events in buckets covering specific time ranges, then the summaries can be generated on a bucket-by-bucket basis. Note that producing intermediate summaries can save the work involved in re-running the query for previous time periods, so only the newer event data needs to be processed while generating an updated report. These report acceleration techniques are described in more detail in U.S. Pat. No. 8,589,403, issued on Nov. 19, 2013, and U.S. Pat. No. 8,412,696, issued on Apr. 2, 2011.

The SPLUNK® ENTERPRISE platform provides various schemas, dashboards and visualizations that make it easy for developers to create applications to provide additional capabilities. One such application is the SPLUNK® APP FOR ENTERPRISE SECURITY, which performs monitoring and

alerting operations and includes analytics to facilitate identifying both known and unknown security threats based on large volumes of data stored by the SPLUNK® ENTERPRISE system. This differs significantly from conventional Security Information and Event Management (SIEM) systems that lack the infrastructure to effectively store and analyze large volumes of security-related event data. Traditional SIEM systems typically use fixed schemas to extract data from pre-defined security-related fields at data ingestion time, wherein the extracted data is typically stored in a relational database. This data extraction process (and associated reduction in data size) that occurs at data ingestion time inevitably hampers future incident investigations, when all of the original data may be needed to determine the root cause of a security issue, or to detect the tiny fingerprints of an impending security threat.

In contrast, the SPLUNK® APP FOR ENTERPRISE SECURITY system stores large volumes of minimally processed security-related data at ingestion time for later retrieval and analysis at search time when a live security threat is being investigated. To facilitate this data retrieval process, the SPLUNK® APP FOR ENTERPRISE SECURITY provides pre-specified schemas for extracting relevant values from the different types of security-related event data, and also enables a user to define such schemas.

The SPLUNK® APP FOR ENTERPRISE SECURITY can process many types of security-related information. In general, this security-related information can include any information that can be used to identify security threats. For example, the security-related information can include network-related information, such as IP addresses, domain names, asset identifiers, network traffic volume, uniform resource locator strings, and source addresses. (The process of detecting security threats for network-related information is further described in U.S. patent application Ser. No. 13/956,252, and Ser. No. 13/956,262.) Security-related information can also include endpoint information, such as malware infection data and system configuration information, as well as access control information, such as login/logout information and access failure notifications. The security-related information can originate from various sources within a data center, such as hosts, virtual machines, storage devices and sensors. The security-related information can also originate from various sources in a network, such as routers, switches, email servers, proxy servers, gateways, firewalls and intrusion-detection systems.

During operation, the SPLUNK® APP FOR ENTERPRISE SECURITY facilitates detecting so-called "notable events" that are likely to indicate a security threat. These notable events can be detected in a number of ways: (1) an analyst can notice a correlation in the data and can manually identify a corresponding group of one or more events as "notable;" or (2) an analyst can define a "correlation search" specifying criteria for a notable event, and every time one or more events satisfy the criteria, the application can indicate that the one or more events are notable. An analyst can alternatively select a pre-defined correlation search provided by the application. Note that correlation searches can be run continuously or at regular intervals (e.g., every hour) to search for notable events. Upon detection, notable events can be stored in a dedicated "notable events index," which can be subsequently accessed to generate various visualizations containing security-related information. Also, alerts can be generated to notify system operators when important notable events are discovered.

The SPLUNK® APP FOR ENTERPRISE SECURITY provides various visualizations to aid in discovering security

threats, such as a "key indicators view" that enables a user to view security metrics of interest, such as counts of different types of notable events. For example, FIG. **10A** illustrates an exemplary key indicators view **700** that comprises a dashboard, which can display a value **701**, for various security-related metrics, such as malware infections **702**. It can also display a change in a metric value **703**, which indicates that the number of malware infections increased by 63 during the preceding interval. Key indicators view **700** additionally displays a histogram panel **704** that displays a histogram of notable events organized by urgency values, and a histogram of notable events organized by time intervals. This key indicators view is described in further detail in pending U.S. patent application Ser. No. 13/956,338 filed Jul. 31, 2013.

These visualizations can also include an "incident review dashboard" that enables a user to view and act on "notable events." These notable events can include: (1) a single event of high importance, such as any activity from a known web attacker; or (2) multiple events that collectively warrant review, such as a large number of authentication failures on a host followed by a successful authentication. For example, FIG. **10B** illustrates an exemplary incident review dashboard **710** that includes a set of incident attribute fields **711** that, for example, enables a user to specify a time range field **712** for the displayed events. It also includes a timeline **713** that graphically illustrates the number of incidents that occurred in one-hour time intervals over the selected time range. It additionally displays an events list **714** that enables a user to view a list of all of the notable events that match the criteria in the incident attributes fields **711**. To facilitate identifying patterns among the notable events, each notable event can be associated with an urgency value (e.g., low, medium, high, critical), which is indicated in the incident review dashboard. The urgency value for a detected event can be determined based on the severity of the event and the priority of the system component associated with the event. The incident review dashboard is described further in "http://docs.splunk.com/Documentation/PCI/2.1.1/User/IncidentReviewdashboard."

As mentioned above, the SPLUNK® ENTERPRISE platform provides various features that make it easy for developers to create various applications. One such application is the SPLUNK® APP FOR VMWARE®, which performs monitoring operations and includes analytics to facilitate diagnosing the root cause of performance problems in a data center based on large volumes of data stored by the SPLUNK® ENTERPRISE system.

This differs from conventional data-center-monitoring systems that lack the infrastructure to effectively store and analyze large volumes of performance information and log data obtained from the data center. In conventional data-center-monitoring systems, this performance data is typically pre-processed prior to being stored, for example by extracting pre-specified data items from the performance data and storing them in a database to facilitate subsequent retrieval and analysis at search time. However, the rest of the performance data is not saved and is essentially discarded during pre-processing. In contrast, the SPLUNK® APP FOR VMWARE® stores large volumes of minimally processed performance information and log data at ingestion time for later retrieval and analysis at search time when a live performance issue is being investigated.

The SPLUNK® APP FOR VMWARE® can process many types of performance-related information. In general, this performance-related information can include any type of performance-related data and log data produced by virtual machines and host computer systems in a data center. In addition to data obtained from various log files, this performance-related information can include values for performance metrics obtained through an application programming interface (API) provided as part of the vSphere Hypervisor™ system distributed by VMware, Inc. of Palo Alto, Calif. For example, these performance metrics can include: (1) CPU-related performance metrics; (2) disk-related performance metrics; (3) memory-related performance metrics; (4) network-related performance metrics; (5) energy-usage statistics; (6) data-traffic-related performance metrics; (7) overall system availability performance metrics; (8) cluster-related performance metrics; and (9) virtual machine performance statistics. For more details about such performance metrics, please see U.S. patent Ser. No. 14/167,316 filed 29 Jan. 2014, which is hereby incorporated herein by reference. Also, see "vSphere Monitoring and Performance," Update 1, vSphere 5.5, EN-001357-00, http://pubs.vmware.com/vsphere-55/topic/com.vmware.ICbase/PDF/vsphere-esxi-vcenter-server-551-monitoring-performance-guide.pdf.

To facilitate retrieving information of interest from performance data and log files, the SPLUNK® APP FOR VMWARE® provides pre-specified schemas for extracting relevant values from different types of performance-related event data, and also enables a user to define such schemas.

The SPLUNK® APP FOR VMWARE® additionally provides various visualizations to facilitate detecting and diagnosing the root cause of performance problems. For example, one such visualization is a "proactive monitoring tree" that enables a user to easily view and understand relationships among various factors that affect the performance of a hierarchically structured computing system. This proactive monitoring tree enables a user to easily navigate the hierarchy by selectively expanding nodes representing various entities (e.g., virtual centers or computing clusters) to view performance information for lower-level nodes associated with lower-level entities (e.g., virtual machines or host systems). Exemplary node-expansion operations are illustrated in FIG. **10C**, wherein nodes **733** and **734** are selectively expanded. Note that nodes **731-739** can be displayed using different patterns or colors to represent different performance states, such as a critical state, a warning state, a normal state or an unknown/offline state. The ease of navigation provided by selective expansion in combination with the associated performance-state information enables a user to quickly diagnose the root cause of a performance problem. The proactive monitoring tree is described in further detail in U.S. patent application Ser. No. 14/235,490 filed on 15 Apr. 2014, which is hereby incorporated herein by reference for all possible purposes.

The SPLUNK® APP FOR VMWARE® also provides a user interface that enables a user to select a specific time range and then view heterogeneous data, comprising events, log data and associated performance metrics, for the selected time range. For example, the screen illustrated in FIG. **10D** displays a listing of recent "tasks and events" and a listing of recent "log entries" for a selected time range above a performance-metric graph for "average CPU core utilization" for the selected time range. Note that a user is able to operate pull-down menus **742** to selectively display different performance metric graphs for the selected time range. This enables the user to correlate trends in the performance-metric graph with corresponding event and log data to quickly determine the root cause of a performance problem. This user interface is described in more detail in U.S. patent application Ser. No. 14/167,316 filed on 29 Jan. 2014, which is hereby incorporated herein by reference for all possible purposes.

FIG. 11 illustrates a diagrammatic representation of a computing device 1000 within which a set of instructions for causing the computing device to perform the methods discussed herein may be executed. The computing device 1000 may be connected to other computing devices in a LAN, an intranet, an extranet, and/or the Internet. The computing device 1000 may operate in the capacity of a server machine in client-server network environment. The computing device 1000 may be provided by a personal computer (PC), a set-top box (STB), a server, a network router, switch or bridge, or any machine capable of executing a set of instructions (sequential or otherwise) that specify actions to be taken by that machine. Further, while only a single computing device is illustrated, the term "computing device" shall also be taken to include any collection of computing devices that individually or jointly execute a set (or multiple sets) of instructions to perform the methods discussed herein. In illustrative examples, the computing device 1000 may implement the above described methods 300A-300B for assigning scores to objects based on evaluating triggering conditions applied to datasets produced by search queries.

The example computing device 1000 may include a processing device (e.g., a general purpose processor) 1002, a main memory 1004 (e.g., synchronous dynamic random access memory (DRAM), read-only memory (ROM)), a static memory 1006 (e.g., flash memory and a data storage device 1018), which may communicate with each other via a bus 1030.

The processing device 1002 may be provided by one or more general-purpose processing devices such as a microprocessor, central processing unit, or the like. In an illustrative example, the processing device 1002 may comprise a complex instruction set computing (CISC) microprocessor, reduced instruction set computing (RISC) microprocessor, very long instruction word (VLIW) microprocessor, or a processor implementing other instruction sets or processors implementing a combination of instruction sets. The processing device 1002 may also comprise one or more special-purpose processing devices such as an application specific integrated circuit (ASIC), a field programmable gate array (FPGA), a digital signal processor (DSP), a network processor, or the like. The processing device 1002 may be configured to execute the methods 300A-300B for assigning scores to objects based on evaluating triggering conditions applied to datasets produced by search queries, in accordance with one or more aspects of the present disclosure.

The computing device 1000 may further include a network interface device 1008, which may communicate with a network 1020. The computing device 1000 also may include a video display unit 1010 (e.g., a liquid crystal display (LCD) or a cathode ray tube (CRT)), an alphanumeric input device 1012 (e.g., a keyboard), a cursor control device 1014 (e.g., a mouse) and an acoustic signal generation device 1016 (e.g., a speaker). In one embodiment, video display unit 1010, alphanumeric input device 1012, and cursor control device 1014 may be combined into a single component or device (e.g., an LCD touch screen).

The data storage device 1018 may include a computer-readable storage medium 1028 on which may be stored one or more sets of instructions (e.g., instructions of the methods 300A-300B for assigning scores to objects based on evaluating triggering conditions applied to datasets produced by search queries, in accordance with one or more aspects of the present disclosure) implementing any one or more of the methods or functions described herein. Instructions implementing methods 300A-300B may also reside, completely or at least partially, within main memory 1004 and/or within

processing device 1002 during execution thereof by computing device 1000, main memory 1004 and processing device 1002 also constituting computer-readable media. The instructions may further be transmitted or received over a network 1020 via network interface device 1008.

While computer-readable storage medium 1028 is shown in an illustrative example to be a single medium, the term "computer-readable storage medium" should be taken to include a single medium or multiple media (e.g., a centralized or distributed database and/or associated caches and servers) that store one or more sets of instructions. The term "computer-readable storage medium" shall also be taken to include any medium that is capable of storing, encoding or carrying a set of instructions for execution by the machine and that cause the machine to perform the methods described herein. The term "computer-readable storage medium" shall accordingly be taken to include, but not be limited to, solid-state memories, optical media and magnetic media.

Unless specifically stated otherwise, terms such as "updating," "identifying," "determining," "sending," "assigning," or the like refer to actions and processes performed or implemented by computing devices that manipulate and transform data represented as physical (electronic) quantities within the computing device's registers and memories into other data similarly represented as physical quantities within the computing device memories or registers or other such information storage, transmission or display devices. Also, the terms "first," "second," "third," "fourth," etc. as used herein are meant as labels to distinguish among different elements and may not necessarily have an ordinal meaning according to their numerical designation.

Examples described herein also relate to an apparatus for performing the methods described herein. This apparatus may be specially constructed for the required purposes, or it may comprise a general purpose computing device selectively programmed by a computer program stored in the computing device. Such a computer program may be stored in a computer-readable non-transitory storage medium.

The methods and illustrative examples described herein are not inherently related to any particular computer or other apparatus. Various general purpose systems may be used in accordance with the teachings described herein, or it may prove convenient to construct more specialized apparatus to perform the required method operations. The required structure for a variety of these systems will appear as set forth in the description above.

The above description is intended to be illustrative, and not restrictive. Although the present disclosure has been described with references to specific illustrative examples, it will be recognized that the present disclosure is not limited to the examples described. The scope of the disclosure should be determined with reference to the following claims, along with the full scope of equivalents to which the claims are entitled.

What is claimed is:

1. A method, comprising:
   accessing, by one or more processing devices, a set of events, wherein each event in the set of events is associated with a time stamp and includes a portion of machine data indicative of performance or operation of an information technology environment;
   accessing an object-scoring rule that (i) includes a search query that determines when events meet a triggering condition; (ii) identifies an object representing a component of the information technology environment, an application running in the information technology environment, or a person using a component in the information technology environment, and (iii) specifies a

numerical contribution to a score for the object, the numerical contribution to be applied to the score based at least on part on a determination that the triggering condition is met;

executing the search query of the object-scoring rule against the set of events to determine if the triggering condition of the object-scoring rule is met;

based on determining that the triggering condition is met, generating a record of the numerical contribution specified in the object-scoring rule, the record associating the numerical contribution with a time indicator and indicating the object whose score should be affected by the contribution;

identifying, using one or more records of numerical contributions, a set of numerical contributions having associated time indicators falling within a defined time period; and

calculating the score for the object based on the set of numerical contributions, wherein the score indicates at least one of: an indication of a security risk posed by the component or person that the object represents, an indication of performance of the component of the information technology environment that the object represents, or an indication of performance of the application that the object represents.

2. The method of claim **1**, wherein the portion of machine data in each event comprises raw machine data.

3. The method of claim **1**, wherein the object-scoring rule statically identifies the object whose score should be adjusted when the triggering condition is met.

4. The method of claim **1**, wherein the object-scoring rule variably identifies the object whose score should be adjusted when the triggering condition is met based on a value for a field in one or more particular events that caused the triggering condition to be met, the value for the field derived by applying an extraction rule or regular expression to the portion of machine data in the one or more particular events.

5. The method of claim **1**, wherein the object-scoring rule statically identifies the numerical contribution to be applied to the score of the object.

6. The method of claim **1**, wherein the object-scoring rule variably identifies the numerical contribution to be applied to the score of the object based on a value for a field in one or more particular events that caused the triggering condition to be met, the value for the field derived by applying an extraction rule or regular expression to the portion of machine data in the one or more particular events.

7. The method of claim **1**, wherein the object-scoring rule variably identifies the numerical contribution to be applied to the score of the object based on a number of particular events that caused the triggering condition to be met.

8. The method of claim **1**, wherein the triggering condition includes a value for a field in an event meeting specified criteria.

9. The method of claim **1**, wherein the triggering condition includes an aggregate calculated for a field in events in the set of events meeting specified criteria.

10. The method of claim **1**, wherein the triggering condition includes determining that a threshold number of events meets specified search criteria.

11. The method of claim **1**, further comprising receiving a user request to create a negative score modifier associated with a particular time.

12. The method of claim **1**, further comprising receiving a request to adjust the object's score by a particular positive or negative amount.

13. The method of claim **1**, further comprising causing displaying of a graphical interface enabling a user to generate the object-scoring rule.

14. The method of claim **1**, wherein events in the set of events are derived from log data.

15. The method of claim **1**, wherein events in the set of events are derived from network packet data.

16. The method of claim **1**, further comprising:
causing display of object scores for a plurality of objects.

17. The method of claim **1**, further comprising:
causing display of object scores for a plurality of objects in a descending order of score.

18. The method of claim **1**, wherein executing the search query comprises applying a late binding schema to the set of events, and further comprising:
receiving a selection of a second time period;
determining a second set of numerical contributions, wherein each of the numerical contribution in the second set of numerical contributions has an associated time indicator falling within the second time period, wherein the associated time indicator specifies a time of occurrence of the triggering condition; and
adjusting the score for the object based on an aggregation of the numerical contributions of the second set of numerical contributions.

19. The method of claim **1**, wherein executing the search query comprises applying a late binding schema to the set of events.

20. The method of claim **1**, wherein executing the search query comprises using an extraction rule or regular expression to the portion of machine data in an event to derive a value for a field.

21. The method of claim **1**, wherein the time indicator specifies a time of occurrence of the triggering condition.

22. The method of claim **1**, wherein the numerical contribution is proportional to a number of occurrences of the triggering condition.

23. The method of claim **1**, wherein the numerical contribution is proportional to a metric associated with a certain activity of the component, the person or the application represented by the object.

24. The method of claim **1**, further comprising: sending a message comprising the score for the object.

25. The method of claim **1**, further comprising: executing a shell script having a parameter defined based on the score for the object.

26. The method of claim **1**, further comprising: causing display of at least a portion of the machine data that triggered the object-scoring rule.

27. A computer-readable non-transitory storage medium comprising executable instructions that, when executed by one or more processing devices, cause the processing devices to perform operations comprising:
accessing, by the processing devices, a set of events, wherein each event in the set of events is associated with a time stamp and includes a portion of machine data indicative of performance or operation of an information technology environment;
accessing an object-scoring rule that (i) includes a search query that determines when events meet a triggering condition; (ii) identifies an object representing a component of the information technology environment, an application running in the information technology environment, or a person using a component in the information technology environment, and (iii) specifies a numerical contribution to a score for the object, the

numerical contribution to be applied to the score based at least on part on a determination that the triggering condition is met;

executing the search query of the object-scoring rule against the set of events to determine if the triggering condition of the object-scoring rule is met;

based on determining that the triggering condition is met, generating a record of the numerical contribution specified in the object-scoring rule, the record associating the numerical contribution with a time indicator and indicating the object whose score should be affected by the contribution;

identifying, using one or more records of numerical contributions, a set of numerical contributions having associated time indicators falling within a defined time period; and

calculating the score for the object based on the set of numerical contributions, wherein the score indicates at least one of: an indication of a security risk posed by the component or person that the object represents, an indication of performance of the component of the information technology environment that the object represents, or an indication of performance of the application that the object represents.

28. The computer-readable non-transitory storage medium of claim 27, wherein the object-scoring rule variably identifies the object whose score should be adjusted when the triggering condition is met based on a value for a field in one or more particular events that caused the triggering condition to be met, the value for the field derived by applying an extraction rule or regular expression to the portion of machine data in the one or more particular events.

29. A computer system, comprising:

a memory; and

one or more processing devices, coupled to the memory, to:

access a set of events, wherein each event in the set of events is associated with a time stamp and includes a portion of machine data indicative of performance or operation of an information technology environment;

access an object-scoring rule that (i) includes a search query that determines when events meet a triggering

condition; (ii) identifies an object representing a component of the information technology environment, an application running in the information technology environment, or a person using a component in the information technology environment, and (iii) specifies a numerical contribution to a score for the object, the numerical contribution to be applied to the score based at least on part on a determination that the triggering condition is met;

execute the search query of the object-scoring rule against the set of events to determine if the triggering condition of the object-scoring rule is met;

based on determining that the triggering condition is met, generate a record of the numerical contribution specified in the object-scoring rule, the record associating the numerical contribution with a time indicator and indicating the object whose score should be affected by the contribution;

identify, using one or more records of numerical contributions, a set of numerical contributions having associated time indicators falling within a defined time period; and

calculate the score for the object based on the set of numerical contributions, wherein the score indicates at least one of: an indication of a security risk posed by the component or person that the object represents, an indication of performance of the component of the information technology environment that the object represents, or an indication of performance of the application that the object represents.

30. The computer system of claim 29, wherein the object-scoring rule variably identifies the object whose score should be adjusted when the triggering condition is met based on a value for a field in one or more particular events that caused the triggering condition to be met, the value for the field derived by applying an extraction rule or regular expression to the portion of machine data in the one or more particular events.

* * * * *